

INVENTORS

Sanjeev Gupta

Ravi Shankar

Ajai Kapoor

Prakash Kumar Pati

Corvin Bazgan

Subbarao Nimmakayala

FACILITATION OF MULTI-PROJECT MANAGEMENT USING
CRITICAL CHAIN METHODOLOGY

PARTIAL WAIVER OF COPYRIGHT

[0001] All of the material in this patent application is subject to copyright protection under the copyright laws of the United States and of other countries. As of the first effective filing date of the present application, this material is protected as unpublished material. However, permission to copy this material is hereby granted to the extent that the copyright owner has no objection to the facsimile reproduction by anyone of the patent documentation or patent disclosure, as it appears in the United States Patent and Trademark Office patent file or records, but otherwise reserves all copyright rights whatsoever.

CROSS-REFERENCED APPLICATIONS

[0002] Not applicable.

FIELD OF THE INVENTION

[0003] This invention generally relates to the field of project management and more specifically to efficient multi-project management utilizing critical chain methodology.

DESCRIPTION OF RELATED ART

[0004] In recent years, project management software has gained increasing popularity. Close to \$2 billion is spent on project management software every year. However, it is still the case that most projects, regardless of industry or type, are delivered late, over budget and under scope. Consider the statistics that show 1) over 83% of information technology (IT) projects are delivered late or over-budget, 2) projects completed by large companies have only 42% of the originally designed features and functions, 3) over 85% of engineering projects in the semiconductor industry finish late, 4) on average, high-tech projects are late by 100% despite the use of project management software and traditional tools, 5) 80% of all embedded systems are delivered late and 6) most defense projects take too long and cost too much.

[0005] Currently, a popular approach to project management is the critical path project management methodology. The critical path project management methodology has been utilized in the industry for many years. In brief, the critical path methodology teaches that every project or set of projects has one critical path that must be managed in order to realize established goals. According to the theory, every project has one longest chain of tasks/events, i.e., the critical path, that should be managed in order to maintain time and budget goals. This approach, however, does not allow for the possibility that the critical

path keeps changing due to uncertainties. These uncertainties affect projects adversely and are typically the reason for time and budget overruns for large projects.

[0006] In theory, if you create a good plan and follow it, your projects will be completed on time. In reality, too many uncertainties arise along the way: requirements change, technology fails, vendors do not deliver, work materializes slower than expected, approvals are not granted on time and priorities change. As uncertainties strike, plans go awry.

[0007] Uncertainties can affect projects adversely in several ways. The cascade effect propagates delays but does not allow for the realization of gains. Projects lose time because either all of the preceding activities have not finished, or the needed resources are working elsewhere. Substantial time and capacity are lost as a result. Additionally, resources go idle waiting for work to arrive. Such idling is often not visible because people can continue to fine-tune already completed work in the meantime.

[0008] Another way in which uncertainties can affect projects adversely is multitasking, whereby people shuttle between tasks, killing productivity and stretching projects. As schedules start slipping, people are needed on multiple projects at once. As people shuttle between tasks without finishing one at a time (multitasking), this results in duration stretch, whereby each task takes longer, switching costs, whereby extra capacity is needed for task set-up and set-down, and concentration lapse, whereby quality suffers when people cannot concentrate on one task at a time.

[0009] Yet another way in which uncertainties can affect projects adversely is human behavior, where safety is hidden in commitments. Having experienced constant

delays, workers quickly learn to hide safeties in their estimates. However, these safeties are invariably wasted because of procrastination. With safeties embedded in commitments, it only becomes more tempting to take a slow start. Safeties are also wasted because of a lack of reporting early finishes. People do not report early finishes because they are afraid that the next time, these early finishes will become hard expectations.

[0010] A new approach to project management, the critical chain project management methodology, offers a solution to manage uncertainties in a single project environment, where resources are dedicated to individual projects. A key component to this approach suggests putting blocks of unscheduled time, called buffers, at key integration points in project plans to absorb the shocks of uncertainties. Another key component of the critical chain project management methodology suggests that buffers are managed during execution to ensure the project meets established time goals. A description of the critical chain-based project management method was first published in *Critical Chain*, by Eliyahu M. Goldratt, North River Press 1997.

[0011] This new project management approach, however, has been very difficult to implement in multi-project environments. This is because resources in a multi-project environment are not just dedicated to one project, but instead are shared across multiple projects. In a multi-project environment, buffer management based on a single project fails to work. In this environment, task priorities are needed not just within one project, but among multiple projects. Furthermore, most multi-project teams are located in multiple locations and are managed by individual project managers. These disparate teams, however, draw upon a common pool of resources on a timely basis. For this reason,

implementing the critical chain project management methodology in a multi-project environment requires multi-project buffer management. Specifically, multi-project buffer management is needed in order to provide task priorities across multiple projects. Current project management software lacks multi-project buffer management and task priorities across multiple projects, hampering an organization's ability to deliver multiple projects on time and within budget.

[0012] One common problem with existing project management software based on critical chain is the lack of task prioritization across projects. Task prioritization refers to the calculation of priority for each task. Task priorities are important during execution since task managers must be able to identify, at any given time, those tasks that are most critical to the timely completion of the project. Existing project management packages based on critical chain calculate task priority within a project using a simple formula that merely reports the status of buffers, and leaves it up to the project manager to figure out what to do as a result. This is inadequate in large multi-project environments since true task priorities should include priorities across multiple projects, and should be based on multiple factors such as relative project priority, relative buffer priority, buffer consumption rates and other factors.

[0013] Without the calculation of task priorities across multiple projects, managers are unable to see the Big Picture and hence properly prioritize people and resources to the correct tasks. The calculation of a task priority for a single project shows only a partial priority of a particular task. The calculation of task priorities across multiple projects, however, allows managers to see a complete picture of the entire project and thus see the true priority of a particular task.

[0014] Another common problem with existing project management software based on critical chain is lack of online buffer management. This is inadequate as it would be beneficial for all buffer management related functions to be performed over the Web in a client-server environment, wherein, for example, a project manager could check out the project plan from the central database via the Web, modify the project and then submit the project back to the database using the Web. This would eliminate the need for a project manager to rely on somebody else to do perform these tasks, and the delays associated with them. It would be further beneficial for execution processes to occur over the Web, such as the updating of task manager statuses and the submission of completed work. Additionally, it would be beneficial for buffer management settings to be configured over the Web, eliminating the need for an administrator to sit at a particular workstation. Lastly, it would be beneficial for buffer management reports to be viewed easily over a network. This would allow a wider audience greater access to status and progress information.

[0015] Therefore a need exists to overcome the problems discussed above, and particularly for a way to facilitate multi-project management using critical chain methodology to more efficiently deal with uncertainties.

SUMMARY OF THE INVENTION

[0016] The present invention, according to a preferred embodiment, overcomes problems with the prior art by providing multi-project buffer management for an efficient and easy-to-implement multi-project management system utilizing the critical chain methodology.

[0017] The embodiments of the present invention allow for the efficient calculation of adequate task prioritization for a single project and across multiple projects, wherein the calculation for task priority takes into account relative project priorities, relative buffer priorities and relative buffer consumption rates. The embodiments of the present invention further allow for an extended network (Internet or Web, for example) interface for the buffer management processes.

[0018] In an embodiment of the present invention, a method on a computer for providing critical chain-based project management across a plurality of projects is disclosed. The method includes generating a plurality of plans, each of the plurality of plans corresponding to one of the plurality of projects, wherein a project comprises at least one task. The method further includes generating buffers for each of the plurality of projects and reconciling project resources among the plurality of projects. The method further includes executing the plurality of project plans, including allowing a user to manage the buffers across the plurality of projects. The user is further provided with information associated with buffers for the plurality of projects, so as to evaluate the status of the plurality of projects. Additionally, the user is provided with task prioritization for any task of the plurality of projects, wherein task prioritization is calculated across the plurality of projects.

[0019] In another embodiment of the present invention, a network interface is available to the user for allowing the user to manage the buffers across the plurality of projects, providing the user with information associated with buffers for the plurality of projects and providing the user with task prioritization for any task of the plurality of projects. The network interface can be a Web or Internet interface.

including allowing a user to manage the buffers across the plurality of projects. The user is further provided with information associated with buffers for the plurality of projects, so as to evaluate the status of the plurality of projects. Additionally, the user is provided with task prioritization for any task of the plurality of projects, wherein task prioritization is calculated across the plurality of projects.

[0022] The foregoing and other features and advantages of the present invention will be apparent from the following more particular description of the preferred embodiments of the invention, as illustrated in the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

[0023] The subject matter, which is regarded as the invention, is particularly pointed out and distinctly claimed in the claims at the conclusion of the specification. The foregoing and other features and also the advantages of the invention will be apparent from the following detailed description taken in conjunction with the accompanying drawings.

[0024] FIG. 1 is a block diagram illustrating the client-server architecture of one embodiment of the present invention.

[0025] FIG. 2 is a more detailed block diagram illustrating the overall architecture of one embodiment of the present invention.

[0026] FIG. 3 is an operational flow diagram showing the overall project management process according to one embodiment of the present invention.

[0020] In yet another embodiment of the present invention, a computer system for providing critical chain-based project management across a plurality of projects is disclosed. The computer system includes a client module for generating a plurality of plans, each of the plurality of plans corresponding to one of the plurality of projects, wherein a project comprises at least one task. The computer system further includes a buffer module for generating buffers for each of the plurality of projects and a reconciliation module for reconciling project resources among the plurality of projects. The computer system further includes an execution module for executing the plurality of project plans, comprising an interface for allowing a user to manage the buffers across the plurality of projects. The interface further provides to the user information associated with buffers for the plurality of projects, so as to evaluate the status of the plurality of projects, and task prioritization for any task of the plurality of projects, wherein task prioritization is calculated across the plurality of projects. The interface is provided over a network, such as a WAN.

[0021] In yet another embodiment of the present invention, a computer readable medium for providing critical chain-based project management across a plurality of projects is disclosed. The computer readable medium includes instructions for generating a plurality of plans, each of the plurality of plans corresponding to one of the plurality of projects, wherein a project comprises at least one task. The computer readable medium further includes instructions for generating buffers for each of the plurality of projects and reconciling project resources among the plurality of projects. The computer readable medium further includes instructions for executing the plurality of project plans, including allowing a user to manage the buffers across the plurality of projects. The computer readable medium further includes instructions for executing the plurality of project plans,

[0027] FIG. 4 is a block diagram illustrating the planning phase of the project management process according to one embodiment of the present invention.

[0028] FIG. 5 is an operational flow diagram showing the planning phase of the project management process according to one embodiment of the present invention.

[0029] FIG. 6 is a screenshot of a graphic showing a project plan.

[0030] FIG. 7 is a screenshot of a graphic showing a project plan with buffers in one embodiment of the present invention.

[0031] FIG. 8 is a screenshot of a pipelining user interface for date reconciliation in one embodiment of the present invention.

[0032] FIG. 9 is a screenshot of a pipelining user interface for resource reconciliation in one embodiment of the present invention.

[0033] FIG. 10 is a block diagram illustrating the execution phase of the project management process according to one embodiment of the present invention.

[0034] FIG. 11 is a screenshot of a task updating user interface in one embodiment of the present invention.

[0035] FIG. 12 is an illustration of a project status graph report in one embodiment of the present invention.

[0036] FIG. 13A is a screenshot of a first task prioritization user interface in one embodiment of the present invention.

[0037] FIG. 13B is a screenshot of a second task prioritization user interface in one embodiment of the present invention.

[0038] FIG. 14 is an operational flow diagram showing the task prioritization calculation process according to one embodiment of the present invention.

[0039] FIG. 15 is a block diagram of an information processing system useful for implementing the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0040] It should be understood that these embodiments are only examples of the many advantageous uses of the innovative teachings herein. In general, statements made in the specification of the present application do not necessarily limit any of the various claimed inventions. Moreover, some statements may apply to some inventive features but not to others. In general, unless otherwise indicated, singular elements may be in the plural and vice versa with no loss of generality. In the drawing like numerals refer to like parts through several views.

[0041] The present invention, according to a preferred embodiment, overcomes problems with the prior art by providing multi-project buffer management for an efficient and easy-to-implement multi-project management system utilizing the critical chain methodology.

[0042] One advantage of the present invention is the calculation of task priorities while taking a multitude of factors into account, such as buffer priorities, the amount of buffer

consumed by a task, chain lengths, maximum buffer consumption, the amount of the current chain that has been completed, amount of buffer recovered, and other factors. Another advantage of the present invention is the calculation of task priorities among multiple projects. Implementing the critical chain project management methodology in a multi-project environment requires multi-project buffer management. Thus, the present invention provides multi-project buffer management in order to provide task priorities across multiple projects. The calculation of task priorities across multiple projects allows managers to assign resources based on the overall need of all projects.

[0043] Yet another advantage of the present invention is online buffer management. It is beneficial for all buffer management related functions to be performed over the Web in a client-server environment, such as a project manager submitting a project over the Web. This eliminates the need for a project manager to create a project at a particular computer. It is further beneficial as execution processes can occur over the Web, such as the updating of task manager statuses and the submission of completed work. Additionally, it is beneficial for buffer management settings to be configured over the Web, eliminating the need for an administrator to sit at a particular workstation. Lastly, it is beneficial for buffer management reports to be viewed easily over a network. This allows a wider audience greater access to status and progress information.

OVERVIEW

[0044] FIG. 1 is a block diagram illustrating the client-server architecture of one embodiment of the present invention. The exemplary embodiments of the present

invention adhere to the system architecture of FIG. 1. FIG. 1 shows an embodiment of the present invention wherein clients interact with a critical chain-based project management system over a network, such as in an enterprise implementation of the project management system that services multiple users in more than one location and for multiple projects. FIG. 1 shows client computers 106 through 108 connected to a network 104. It should be noted that although FIG. 1 shows only two client computers 106 and 108, the system of the present invention supports any number of client computers. FIG. 1 also shows a critical chain-based project management server system 102 connected to the network 104. The project management server system 102 is described in more detail with reference to FIG. 3 below.

[0045] In an embodiment of the present invention, the computer systems of client computers 106 through 108 and server system 102 are one or more Personal Computers (PCs), Personal Digital Assistants (PDAs), hand held computers, palm top computers, laptop computers, smart phones, game consoles or any other information processing devices. A PC can be one or more IBM or compatible PC workstations running a Microsoft Windows or LINUX operating system, one or more Macintosh computers running a Mac OS operating system, or an equivalent. In another embodiment, the client computers 106 through 108 and server system 102 are a server system, such as SUN Ultra workstations running a SunOS operating system or IBM RS/6000 workstations and servers running the AIX operating system. The computer systems of computers 106 through 108 and server system 102 are described in greater detail below with reference to FIG. 15.

[0046] In an embodiment of the present invention, the network 104 is a circuit switched network, such as the Public Service Telephone Network (PSTN). In another embodiment, the network 104 is a packet switched network. The packet switched network is a wide area network (WAN), such as the global Internet, a private WAN, a local area network (LAN), a telecommunications network or any combination of the above-mentioned networks. In yet another embodiment, the structure of the network 104 is a wired network, a wireless network, a broadcast network or a point-to-point network.

[0047] FIG. 2 is a more detailed block diagram illustrating the overall architecture of one embodiment of the present invention. FIG. 2 shows more detail regarding the client-server nature of the present invention, as shown in FIG. 1.

[0048] FIG. 2 shows the components of the critical chain-based project management server system 102 connected to the network 104. The project management server system 102 includes a database 206, a database management system 204 and a project management application 202. The database 206 is a repository for data used by project management application 202 during the course of a project or projects. The database 206 includes all information necessary for performing the functions of the project management server system 102, including project planning, project execution, buffering, pipelining, task updating, etc. These functions are described in greater detail below. The database 206 also includes project setting information, user information, reports, and other information that is modified or accessed by the project management application 202 during the course of a project or projects. Database 206 can be any commercially database, such as an Oracle Database, Enterprise or Personal Edition, available from Oracle Corporation, or a Microsoft SQL Server or Access 2000 database available from Microsoft Corporation.

[0049] Database management system 204 is an application that controls the organization, storage and retrieval of data (fields, records and files) in database 206. The database management system 204 accepts requests for data from the project management application program 202 and instructs the operating system to transfer the appropriate data. Database management system 204 may also control the security and integrity of the database 206. Data security prevents unauthorized users from viewing or updating certain portions of the database 206. Database management system 204 can be any commercially database management system, such as the Oracle E-Business Suite available from Oracle Corporation.

[0050] FIG. 2 also shows the project management application 202, which performs substantially the functions of the present invention, as described in greater detail below. Project management application 202 connects directly to the network 104 via a network interface, such as a network interface card. Optionally, the project management server system 102 includes a Web server that connects to the network 104 via a network interface. The project management application 202 is logically connected to the Web server, which provides a Web interface available to clients (such as clients 106 through 108) of the project management server system 102. This option is advantageous as a Web interface allows any clients having a Web browser to connect to the project management server system 102. A Web interface provides a simple, efficient, highly compatible, economical and highly available connection to the project management server system 102 to a wide range of clients.

[0051] FIG. 2 further shows a group of clients for connecting to the project management server system 102. These clients include a single project-planning client 210, a buffering

client 214, and a resource reconciliation client 216, otherwise known as a pipelining client 216. The single project planning client 210 allows a project manager to generate and submit a project plan to the project management server system 102. The buffering client 214 modifies a project plan to include a required set of buffers. The resource reconciliation client 216 processes all project plans and reconciles all resource conflicts in order to produce a master project plan. The resource reconciliation client 216, or pipelining client, performs cross-project or multiple project planning, as opposed to single project planning.

[0052] Single project planning client 210, the buffering client 214, and the resource reconciliation client 216 are client applications, such as C++, Visual Basic, Java applet, a Java scriptlet, Java script, Perl script, an Active X control or any self-sufficient application executing on a client computer. The clients of FIG. 2 can communicate with the project management server system 102 via a Web interface such as a commercially available Web browser, e.g., Netscape Navigator and Microsoft Internet Explorer. The functions of the single project planning client 210, the buffering client 214, and the resource reconciliation client 216 are described in greater detail below.

[0053] FIG. 2 further shows a settings module 210 for configuring the settings for the project management server system 102. Settings module 210 can be used to manage user and group accounts, manage user and group permissions and authorizations, set use policies, set system and Web interface parameters, modify configuration settings on the project management server system 102 and the like. Settings module 210 is a client application, such as an application programmed in C++, Visual Basic, a Java applet, a Java scriptlet, Java script, Perl script, an Active X control or any self-sufficient application

executing on a client computer. The settings module 210 can communicate with the project management server system 102 via a Web interface such as a commercially available Web browser. Alternatively, the settings module 210 is a simple Web interface realized through a commercially available Web browser, wherein the user enters and/or modifies settings via a Web page.

[0054] FIG. 2 further shows a task-updating module 218, used by task managers to update the status of tasks and projects with the project management server system 102 during the course of a project or projects. The task-updating module 218 can be used to manage the creation and modification of task information for projects, before and during execution. The task-updating module 218 can further be used to update task statuses, forecast how much more time it will take to finish the task, modify task attributes, provide information on what may be holding up a task, and the like. The task updating module 218 is a client application, such as an application programmed in HTML or XML, a Java applet, a Java scriptlet, Java script, Perl script, an Active X control or any self-sufficient application executing on a client computer. The task-updating module 218 can communicate with the project management server system 102 via a Web interface such as a commercially available Web browser. Alternatively, the task-updating module 218 is a simple Web interface realized through a commercially available Web browser, wherein the user enters and/or modifies task information via a Web page. The functions of the task-updating module 218 are described in greater detail below.

[0055] FIG. 2 further shows a reports module 220, used by task and project managers to evaluate the status of a project or projects with the project management server system 102 during the course of a project or projects. The reports module 220 can be used to

download various kinds of reports, such as progress chart and graphs, which can be used to evaluate the status of a project during execution. The reports module 220 is a client application, such as a Java applet, a Java scriptlet, Java script, Perl script, an Active X control, an application programmed in HTML or XML or any self-sufficient application executing on a client computer. The reports module 220 can communicate with the project management server system 102 via a Web interface such as a commercially available Web browser. Alternatively, the reports module 220 is a simple Web interface realized through a commercially available Web browser, wherein the user selects reports for download via a Web page.

[0056] It should be noted that in the embodiment of the present invention described above, the modules 210, 218 and 220 and the clients 212, 214 and 216 are depicted as separate from the project management server system 102. In this embodiment, the modules 210, 218 and 220 and the clients 212, 214 and 216 communicate with the computer system of the project management server system 102 over a network 104 or other communication medium. In an alternative embodiment of the present invention, any one or all of the modules 210, 218 and 220 and the clients 212, 214 and 216 can be integrated with the computer system of the project management server system 102. In this alternative embodiment, those modules or clients that are integrated with the project management server system 102 share the same resources as the project management server system 102.

[0057] FIG. 3 is an operational flow diagram showing the overall project management process according to one embodiment of the present invention. FIG. 3 shows the three main phases of any project management process: the planning phase, the execution

phase and the post-execution analysis phase. The operation and control flow of FIG. 3 begins with step 302 and proceeds directly to step 304.

[0058] In step 304, the pre-execution planning phase includes such functions as configuring settings, generating and submitting single projects, buffering of projects, multiple project planning, and other functions. Functions of the planning phase are described in more detail with reference to FIGs. 4 and 5 below. The pre-execution planning phase further includes a portfolio optimization function that allows high-level project planning. The portfolio optimization function calculates those project objectives that can be met with the available project resources. Conversely, portfolio optimization function can calculate the project resources necessary to meet those project objectives that are most desirable.

[0059] In step 306, the execution phase of the project management process includes the functions of buffer management, task updating, report generation and other functions. The functions of the execution phase are described in more detail with reference to FIGs. 6 and 7 below. Lastly, in step 308, the post-execution analysis phase includes a performance diagnostic function wherein project managers and other administrator can review the execution of the project or projects. The performance diagnostic function can be used to determine problem items, such as bottlenecks, or buffer-consuming tasks/projects, and to determine resource deficiencies of a project or projects. In step 310, the operational flow diagram of FIG. 3 stops.

PRE-EXECUTION PLANNING

[0060] FIG. 4 is a block diagram illustrating the planning phase of the project management process according to one embodiment of the present invention. FIG. 4 shows more detail of the pre-execution planning phase of step 304 of FIG. 3, including the functions of configuring settings, generating and submitting single projects, buffering of projects, multiple project planning, and other functions. FIG. 4 begins with the generation of a project plan by the single project-planning client 212.

[0061] In FIG. 4, the single project-planning client 212 accesses a project template 404 from a project template bank 402. A project template 404 is a template of a project plan, defined beforehand such that projects of similar type and/or scope need not be generated from scratch by a project manager. Project templates can be re-used among project managers in order to save time and resources. The project template bank 402 is a database or repository of project templates. In an embodiment of the present invention, the project template bank 402 can be any commercially database, such as an Oracle Database.

[0062] A project manager interacts with the single project-planning client 212 in order to generate a project plan 408 based on the project template 404 from the project template bank 402. The project plan 408 is then submitted to the project management system 102, for example, over the Web via a Web page or over the Internet via an email message. A project plan 408 is map, table, chart or any other organization of information that shows the intended progression of a project from start to finish. In one embodiment of the present invention, Microsoft Project, a project management application available from Microsoft Corporation, is used as the single project-planning client 212 to create a project plan. In this embodiment, Microsoft Project, which does not utilize a critical chain-based

project management methodology, is used in conjunction with a plug-in, API or standalone application that offers critical chain-based calculations for generating a project plan.

[0063] FIG. 6 is a screenshot of a graphic showing a project plan. FIG. 6 shows an example of a graphic depicting a project plan 402 generated using Microsoft Project. The graphic of FIG. 6 shows chronological dates along the top axis of the graphic 602. The project plan comprises individual tasks, shown as task blocks 604, 606 and 608 in the graphic. The length of each task block dictates the amount of time that is required to complete each task. Also shown are connecting lines, such as lines 610 and 612, which connect task blocks 604 and 606, as well as 606 and 608. These connecting lines show the interdependence of task blocks, such as the interdependence of task blocks 604 and 606. As depicted, the completion of task 606 depends on the completion of task 604.

[0064] Returning to FIG. 4, the generation of a project plan 408 by the single project-planning client 212 gives way to the buffering client 214, which inserts buffers into the project plan 408. The critical chain-based methodology of Goldratt (see bibliography above) teaches the use of various types of buffers that are used to protect time deadlines and budget goals in the event of uncertainties during project execution. Four types of buffers are used: project buffers located at the end of a project, feeding buffers located at the point at which one set of tasks or projects feeds into another set of tasks or projects, milestone buffers that are located at project milestones and project-to-project (or P2P) buffers that are located at points where one project feeds into another project. The buffering client 214 processes the project plan 408 and inserts the aforementioned buffers into the project plan 408, resulting in a project plan with buffers 406.

[0065] FIG. 7 is a screenshot of a graphic showing a project plan with buffers in one embodiment of the present invention. FIG. 7 shows an example of a graphic depicting a project plan with buffers 406 generated using Microsoft Project. The graphic of FIG. 7 shows chronological dates along the top axis of the graphic 702. The project plan comprises individual tasks, shown as task blocks in the graphic. The length of each task block dictates the amount of time that is required to complete each task. Also shown are connecting lines that connect task blocks. These connecting lines show the interdependence of task blocks. FIG. 7 also shows examples of feeding buffers, 704, 706, 708, 710 and 712. Each of the feeding buffers is located at the end of a set of tasks that are being fed into another set of tasks. FIG. 7 also shows an example of a project buffer 715. The project buffer 715 is located at the end of the entire project, thus protecting the predicted finish for the entire project.

[0066] Returning to FIG. 4, the generation of a project plan with buffers 406 by the buffering client 214 gives way to the pipelining client 216, which processes all project plans and reconciles all resource conflicts in order to produce a master project plan 414. The pipelining client 216, or resource reconciliation client, performs cross-project or multiple project planning, as opposed to single project planning. The pipelining client 216 takes as input the project plan with buffers 406, generated by the buffering client 214. The pipelining client 216 also takes an input all other project plans 412 for the entire project, as well as a global resource table 410. The global resource table 410 is a complete set of resources that are available for the completion of all projects. Resources include man hours associated with workers, computing resources, raw materials, equipment, etc.

[0067] The pipelining client 216 performs a variety of functions, one of which is described with reference to FIG. 8. The pipelining client 216 allows a user to set a project completion date for one, more than one or all projects in a set of projects and provide the result of such date constraints on the over project plan. For example, see FIG. 8. FIG. 8 is a screenshot of a pipelining user interface 800 for date reconciliation in one embodiment of the present invention. The user interface 800 shows that a project “puma_02” has been selected from a set of sub-projects of a greater project, as shown in the text field 802. The user may enter a requested completion date of the sub-project “puma_02” into the text field 804. Once a date has been entered, the user may press the “Test Impact” button 806 to test the impact on the other sub-projects and the project as a whole of mandating the requested completion date entered by the user.

[0068] The impact is shown in the window 808, which lists all sub-projects and project data associated with each sub-project. For each sub-project the window 808 shows a variety of information such as the projected date of completion in column 810, the additional lateness of the date of completion in column 812, the cumulative lateness of the date of completion in column 814, and the buffer status in column 816. As the user presses the “Test Impact” button 806 to test the impact of mandating the requested completion date entered by the user, the pipelining client 216 performs calculations to determine the impact of such a mandate on the other sub-projects. Inevitably, as one sub-project is elevated in priority, other sub-projects are downgraded in priority. As a result, as the completion date of the sub-project “puma_02” is mandated, the projected date, or lateness of the other sub-projects may suffer. In this way, a project manager may modify sub-project priorities to reach the optimal combination of completion dates.

[0069] Another function of the pipelining client 216 is described with reference to FIG. 9. The pipelining client 216 allows a user to set resource loads in order to meet time and budget goals. For example, see FIG. 9. FIG. 9 is a screenshot of a pipelining user interface 900 for resource reconciliation in one embodiment of the present invention. The user interface 900 shows a list of resources, such as the diagnostic engineer resource 904, on the bar chart 902 and the load on each resource is shown as the entire horizontal bar adjacent to a resource name, such as bar 906. A load is defined as the percentage of an available resource that is scheduled for use during a project. For example, if a worker is available for working a total of 2,000 man-hours in one year and the project schedules the worker for 1,000 man-hours worth of tasks for the year on the project, then the load on the worker is 50%.

[0070] Whereas the load on each resource is shown as the entire horizontal bar adjacent to a resource name, the red or dark portion of the bar indicates that portion of the resource that is scheduled on tasks that are part of the critical chain. According to the critical chain-based project management methodology, every project or set of projects has one critical chain of tasks/events that must be protected in order to realize established goals. The critical chain should be prioritized in order to maintain time and budget goals.

[0071] FIG. 9 shows that the diagnostic engineer resource 904 has been constrained, as indicated by an icon 908. To constrain means to normalize a resource to be loaded to a percentage below 100%. It may be that before the diagnostic engineer resource 904 was constrained, the resource was loaded to more than 100%. This occurs when the pipelining client 216 produces the resource load chart 902 based on a list of user

objectives, wherein the objectives only took time goals, and not resource constraints, into account. Thus, the purpose of the pipelining client 216 is to reconcile resource constraints among the various sub-projects of a greater project. Once the diagnostic engineer resource 904 has been constrained to a load below 100%, the chart 902 shows that all other resources are also below 100% load. In this way, a project manager may adjust loads and objectives in order to reach the optimal combination of resource loads, objectives and completion dates.

[0072] FIG. 5 is an operational flow diagram showing the planning phase of the project management process according to one embodiment of the present invention. FIG. 5 shows more detail of the planning phase of step 304 of FIG. 3, including the functions of configuring settings, generating and submitting single projects, buffering of projects, multiple project planning, and other functions. FIG. 5 begins with step 502 and proceeds immediately to step 504.

[0073] In step 504, the settings of the project management server system 102 are configured by a user, such as a system administrator or a project manager. The settings module 210 can be used for configuring the settings of the project management server system 102. Settings module 210 can be used to manage user and group accounts, manage user and group permissions and authorizations, set use policies, set system and Web interface parameters, modify configuration settings on the project management server system 102 and the like. As explained above, the settings module 210 can be a simple Web interface realized through a commercially available Web browser, wherein the user enters and/or modifies settings via a Web page.

[0074] In step 506, the single project-planning client 212 accesses a project template 404 from a project template bank 402. A project template 404 is a template of a project plan, defined beforehand such that projects of similar type and/or scope need not be generated from scratch by a project manager. In step 508, a project manager interacts with the single project-planning client 212 in order to generate a project plan 408 based on the project template 404 from the project template bank 402. A project plan 408 is map, table, chart or any other organization of information that shows the intended progression of a project from start to finish. The generation of a project plan 408 by the single project planning client 212 gives way to the buffering client 214, which inserts buffers into the project plan 408 in step 510. The buffering client 214 processes the project plan 408 and inserts a variety of buffers into the project plan 408, resulting in a project plan with buffers 406.

[0075] The generation of a project plan with buffers 406 by the buffering client 214 gives way to the pipelining client 216, which processes all project plans and reconciles all resource conflicts in step 512. The pipelining client 216, or resource reconciliation client, performs cross-project or multiple project planning, as opposed to single project planning. The pipelining client 216 takes as input the project plan with buffers 406, generated by the buffering client 214. The pipelining client 216 also takes an input all other project plans 412 for the entire project, as well as a global resource table 410. The global resource table 410 is a complete set of resources that are available for the completion of all projects. The result of the pipelining client 216 is a master project plan 414, in step 514. In step 516, the operational flow of FIG. 5 stops.

EXECUTION

[0076] Critical chain-based project management methodology includes two main components: the placement of buffers in the correct locations of a project plan and using the status of these buffers during execution to decide what tasks to focus on. This second component is called buffer management. The main idea of buffer management is that the rate at which buffers are depleting is monitored and priority is given to tasks that lie on chains where buffers are fast depleting. In a multi-project environment, however, this simplistic idea is difficult to implement. Project employees often work on multiple projects at the same time, and hence need not only task priorities within a project, but also across projects. This is a complex problem as comparing tasks among different projects is like comparing apples to oranges: some projects have higher strategic priorities, certain milestones in a project that need to be delivered first and individual projects have different amounts of buffer placed in their project plans. This is compounded by the fact that project employees are often located in different places.

[0077] The present invention provides multi-project buffer management that allows organizations to use buffer management in a complex environment, where resources are shared across projects. Task priorities are driven by a combination of relative project priority, relative milestone priority and buffer consumption by each task, requiring a complex set of algorithms. The buffer management process is further Web enabled so that project managers can configure buffering policies over the Web, submit project plans with buffers over the Web, do task updates over the Web, run buffer management functions using the Web, and acquire buffer reports over the Web. Current critical chain-based project management solutions lack the capability of multi-project buffer

management, and only have capability for a single project buffer management and thus at best work only when resources are dedicated to one project.

[0078] FIG. 10 is a block diagram illustrating the execution phase of the project management process according to one embodiment of the present invention. FIG. 10 shows more detail of the execution phase of step 306 of FIG. 3, including the functions of buffer management, task updating, task prioritization, report review and other functions. FIG. 10 shows that the buffer management process 1002 comprises three components, task updating (as performed by task updating module 218), report review (wherein reports are provided by reports module 220) and task prioritization.

[0079] FIG. 10 shows the updating of task statuses via the task-updating module 218. The task-updating module 218 is used by task managers to update the status of tasks and projects with the project management server system 102 during the course of a project or projects. The task-updating module 218 can be used to manage the creation and modification of task information for projects, before and during execution. The task-updating module 218 can further be used to update task statuses, forecast how much more time it will take to finish the task, modify task attributes, provide information on what may be holding up a task, and the like. As explained above, the task updating module 218 can be a simple Web interface realized through a commercially available Web browser, wherein the user enters and/or modifies task information via a Web page.

[0080] The task-updating module 218 performs a variety of functions, one of which is described with reference to FIG. 11. FIG. 11 is a screenshot of a task updating user interface 1100 in one embodiment of the present invention. The user interface 1100 can be realized as a Web page accessible over the Web by a Web browser. The user

interface 1100 shows all tasks for a particular task manager, as indicated in text field 1120, starting on a particular date, as indicated in text field 1122, and for a specified period of time, as indicated in text field 1124. The aforementioned text fields allow for different views of information by selecting various selections in the aforementioned text fields, such as selecting a different task manager in the text field 1120.

[0081] The user interface 1100 shows a list of tasks and, for each task, a task description in column 1106, a task earliest completion date in column 1108, a task status in column 1110, a remaining duration for the task in column 1112 and a last update date in column 1114. As task managers administer the progress of their respective tasks, they can access the user interface 1100 and update certain pieces of information, such as a task status in column 1110 and a remaining duration for the task in column 1112. This information is then used during the execution of the project to determine the status of the project.

[0082] Returning to FIG. 10, also shown is the reports module 220 for the generation and/or download of reports during execution of the project. The reports module 220 is used by task and project managers to evaluate the status of a project or projects of the project management server system 102 during the course of a project or projects. The reports module 220 can be used to download various kinds of reports, such as progress chart and graphs, which can be used to evaluate the status of a project during execution. Examples of chart and graphs that may be downloaded using the reports module 220 are provided below. As explained above, the task updating module 218 can be a simple Web interface realized through a commercially available Web browser, wherein the user enters and/or modifies task information via a Web page.

[0083] FIG. 12 is an illustration of a project status graph report 1200 in one embodiment of the present invention. In an embodiment of the present invention, the project status graph report 1200 can be accessed via a Web page over the Web using a Web browser. The graph 1200 depicts the progress of the project showing % buffer consumed on the y-axis 1210 and % chain complete on the x-axis 1212. The red area 1202 of the graph 1200 indicates that the progress of the project is hampered, the yellow area 1204 of the graph 1200 indicates a warning with regards to the progress of the project and the green area 1206 of the graph 1200 indicates that the progress of the project is occurring as planned. The algorithms of the present invention calculate the current status of the project and plot a point on the graph 1200 representing the current status. The algorithms used to calculate % buffer consumed on the y-axis 1210 and % chain complete on the x-axis 1212 are described in greater detail below.

[0084] Graph 1200 shows six points of reference for the project shown in the graph 1200. Each point marks a place in time in which the progress of the project is plotted. Point 1230 is the first point, marking the beginning of the project. Point 1232 is the second point, showing a setback in the project wherein the % chain complete moved to the negative area and the % buffer consumed has increased. This is due to an instance where the chain upon which the % chain complete calculation is based has increased in length, often due to project uncertainties during project execution. The third point, point 1234, shows that progress has been made toward completing the chain, but that the % buffer consumed has also increased, placing the point 1234 in the red area 1202.

[0085] The fourth point, point 1236, shows that progress has been made toward completing the chain, and that the % buffer consumed has decreased, placing the point

1236 in the yellow area 1204. This shows a buffer gain due to, for example, the early completion of tasks. The fifth point, point 1238, shows that progress has been made toward completing the chain, but that the % buffer consumed has also increased, placing the point 1238 in the yellow area 1204. The sixth point, point 1240, shows that progress has reversed toward completing the chain, and that the % buffer consumed has also increased, placing the point 1240 in the red area 1202. Once again, this may be due to an instance where the chain upon which the % chain complete calculation is based has increased in length, often due to project uncertainties during execution.

[0086] In one example, the % chain complete of the x-axis 1212 is calculated as:

[0087] *% chain complete = (original critical chain duration - remaining duration of the longest chain) / original critical chain duration*

[0088] Original critical chain duration is the time needed to complete the longest chain of tasks including planned availability of resources. The remaining duration of the longest chain is the time needed to finish the longest remaining chain of tasks including planned availability of resources at the current time in the project history.

[0089] In another example, the % chain complete of the x-axis 1212 is calculated differently for milestone chains, wherein a chain of tasks/projects leads to a project milestone, and feeding chains, wherein a chain of tasks/projects feeds into another set of tasks/projects. In this example, the % chain complete of the x-axis 1212 is calculated as:

[0090] *% chain complete = (original duration of longest path including activity and resource dependencies on critical chain - remaining duration of the longest path including*

activity and resource dependencies on critical chain) / original duration of longest path including activity and resource dependencies on critical chain

[0091] This calculation takes into account the fact that chains feeding into milestones often change continuously. To account for this phenomena, the milestone chain includes activity dependencies and only resource dependencies on the critical chain.

[0092] In another example, the % buffer consumed of the y-axis 1210 is calculated by calculating the consumption of the project buffer by the chain feeding into the buffer. If there are multiple chains feeding into the project buffer, the worst value is taken. Since a project buffer can be consumed by any chain of activities and resource dependencies that are linked to this buffer, the % buffer consumed calculation considers the status of the worst chain (which translates into the highest buffer penetration of all chains feeding into the buffer).

[0093] The algorithms of the present invention used to calculate the current status of the project on graph 1200 are advantageous as they take into account the various buffers of the multiple projects and the percentage completed of the multiple projects. This provides a truer representation of the progress of a project, as opposed to viewing only data with respect to one of many projects.

[0094] Returning to FIG. 10, also shown is the provision of task prioritization 1004 during execution of the project. Task prioritization 1004 is a process of the present invention whereby task priorities over multiple projects are calculated. The present invention calculates the task priority for every task of the multitude of projects and thus provides a true multi-project task priority, as the calculation is based on multiple factors such as

relative project priority, relative buffer priority, buffer consumption rates and other factors. The calculation of task priorities across multiple projects allows managers to see a complete picture of the entire project and thus see the true priority of a particular task. The calculation of a task priority for a single project shows only a partial priority of a particular task. The method in which task priorities are calculated is described in more detail with reference to FIG. 14 below.

[0095] FIG. 13A is a screenshot of a first task prioritization user interface 1300 in one embodiment of the present invention. The user interface 1300 can be realized as a Web page accessible over the Web by a Web browser. The user interface 1300 shows all tasks for a particular task manager, as indicated in text field 1320, starting on a particular date, as indicated in text field 1322, and for a specified period of time, as indicated in text field 1324. The aforementioned text fields allow for different views of information by selecting various selections in the aforementioned text fields, such as selecting a different task manager in the text field 1320.

[0096] The user interface 1300 shows a list of tasks and, for each task, a task description in column 1306, a task earliest arrival date in column 1308, a task status in column 1310, a remaining duration for the task in column 1312 and a last update date in column 1314. The user interface 1300 also shows a task priority column 1330 for identifying the priority of the tasks. Priority in column 1330 is shown by color. High priority tasks are shown as red, while low priority tasks are shown as red. A review of the user interface 1300 shows that the first task 1332 is high priority as the task priority in column 1330 is marked as red. The user interface 1300 also shows that the upcoming task 1334 is high priority as the task priority in column 1330 is also marked as red. The method in

which task priorities are calculated is described in more detail with reference to FIG. 14 below.

[0097] As task managers administer the progress of their respective tasks, they can access the user interface 1300 and determine the most important tasks to complete, i.e., those tasks that have the greatest effect on the time and budget goals of the project. This information is then used during the execution of the project to determine where manpower and resources should be allocated.

[0098] FIG. 13B is a screenshot of a second task prioritization user interface 1350 in one embodiment of the present invention. The user interface 1350 can be realized as a Web page accessible over the Web by a Web browser. The user interface 1350 shows all tasks for a particular project manager, as indicated in text field 1370, starting on a particular date, as indicated in text field 1372, and for a specified period of time, as indicated in text field 1374. The aforementioned text fields allow for different views of information by selecting various selections in the aforementioned text fields, such as selecting a different project manager in the text field 1370.

[0099] The user interface 1350 shows a list of tasks and, for each task, a task description in column 1356, a task earliest arrival date in column 1358, a task status in column 1360, a remaining duration for the task in column 1362 and a last update date in column 1364. The user interface 1350 also shows a task priority column 1380 for identifying the priority of the tasks. Priority in column 1380 is shown by color. High priority tasks are shown as red, medium priority tasks are shown as yellow and low priority tasks are shown as green. A review of the user interface 1350 shows that the first four tasks are high priority as the task priority in column 1380 is marked as red for all four. The method

in which task priorities are calculated is described in more detail with reference to FIG. 14 below.

[00100] As project managers administer the progress of their respective tasks, they can access the user interface 1350 and determine the most important tasks to complete, i.e., those tasks that have the greatest effect on the time and budget goals of the project. This information is then used during the execution of the project to determine where manpower and resources should be allocated.

[00101] FIG. 14 is an operational flow diagram showing the task prioritization calculation process according to one embodiment of the present invention. FIG. 14 shows the method in which task priorities are calculated for the task priority columns 1330 and 1380 for FIGs. 13A and 13B. The operation and control flow of FIG. 14 begins with step 1402 and proceeds directly to step 1404.

[00102] In step 1404, for each task, all buffers fed by each task are identified. In other words, all task-buffer pairs are identified. In step 1406, the percent of penetration of each task into a buffer into which it feeds is calculated. I.e., the buffer penetration of each task-buffer pair is calculated. The percent of buffer penetration of any task-buffer pair is equal to the percent of buffer penetration of the chain to which the corresponding task belongs, into the buffer of the task-buffer pair. If the task of the task-buffer pair belongs to multiple chains, the worst buffer penetration value is taken. Also in step 1406, the percent chain length complete for each task-buffer pair is calculated. Note that this calculation takes the percent chain length complete of the longest chain that has one endpoint at the buffer and contains the task.

[00103] In step 1408, for each task-buffer pair, the color of the task with respect to the buffer is calculated, based on the slope setting. In other words, for each task-buffer pair, the color of the task on the graph 1200 of FIG. 12 is calculated. In step 1410, the final color of each task is calculated based on a buffer type priority table that maps priority to colors. The buffer priority table, which may be user defined, defines the color (red, yellow or green) that corresponds to each priority type. In step 1412, the operational flow diagram of FIG. 14 stops.

COMPUTER IMPLEMENTATION

[00104] The present invention can be realized in hardware, software, or a combination of hardware and software in the system described in FIG. 2. A system according to a preferred embodiment of the present invention can be realized in a centralized fashion in one computer system, or in a distributed fashion where different elements are spread across several interconnected computer systems. Any kind of computer system - or other apparatus adapted for carrying out the methods described herein - is suited. A typical combination of hardware and software could be a general-purpose computer system with a computer program that, when being loaded and executed, controls the computer system such that it carries out the methods described herein.

[00105] An embodiment of the present invention can also be embedded in a computer program product, which comprises all the features enabling the implementation of the methods described herein, and which - when loaded in a computer system - is able to carry out these methods. Computer program means or computer program as used in the

present invention indicates any expression, in any language, code or notation, of a set of instructions intended to cause a system having an information processing capability to perform a particular function either directly or after either or both of the following a) conversion to another language, code or, notation; and b) reproduction in a different material form.

[00106] A computer system may include, *inter alia*, one or more computers and at least a computer readable medium, allowing a computer system, to read data, instructions, messages or message packets, and other computer readable information from the computer readable medium. The computer readable medium may include non-volatile memory, such as ROM, Flash memory, Disk drive memory, CD-ROM, and other permanent storage. Additionally, a computer readable medium may include, for example, volatile storage such as RAM, buffers, cache memory, and network circuits. Furthermore, the computer readable medium may comprise computer readable information in a transitory state medium such as a network link and/or a network interface, including a wired network or a wireless network, that allow a computer system to read such computer readable information.

[00107] FIG. 15 is a block diagram of a computer system useful for implementing an embodiment of the present invention. The computer system of FIG. 15 is a more detailed representation of computers 106 through 108 or system 102. The computer system of FIG. 15 includes one or more processors, such as processor 1504. The processor 1504 is connected to a communication infrastructure 1502 (e.g., a communications bus, cross-over bar, or network). Various software embodiments are described in terms of this exemplary computer system. After reading this description, it will become apparent to a

person of ordinary skill in the relevant art(s) how to implement the invention using other computer systems and/or computer architectures.

[00108] The computer system can include a display interface 1508 that forwards graphics, text, and other data from the communication infrastructure 1502 (or from a frame buffer not shown) for display on the display unit 1510. The computer system also includes a main memory 1506, preferably random access memory (RAM), and may also include a secondary memory 1512. The secondary memory 1512 may include, for example, a hard disk drive 1514 and/or a removable storage drive 1516, representing a floppy disk drive, a magnetic tape drive, an optical disk drive, etc. The removable storage drive 1516 reads from and/or writes to a removable storage unit 1518 in a manner well known to those having ordinary skill in the art. Removable storage unit 1518, represents, for example, a floppy disk, magnetic tape, optical disk, etc. which is read by and written to by removable storage drive 1516. As will be appreciated, the removable storage unit 1518 includes a computer usable storage medium having stored therein computer software and/or data.

[00109] In alternative embodiments, the secondary memory 1512 may include other similar means for allowing computer programs or other instructions to be loaded into the computer system. Such means may include, for example, a removable storage unit 1522 and an interface 1520. Examples of such may include a program cartridge and cartridge interface (such as that found in video game devices), a removable memory chip (such as an EPROM, or PROM) and associated socket, and other removable storage units 1522 and interfaces 1520 which allow software and data to be transferred from the removable storage unit 1522 to the computer system.

[00110] The computer system may also include a communications interface 1524. Communications interface 1524 allows software and data to be transferred between the computer system and external devices. Examples of communications interface 1524 may include a modem, a network interface (such as an Ethernet card), a communications port, a PCMCIA slot and card, etc. Software and data transferred via communications interface 1524 are in the form of signals which may be, for example, electronic, electromagnetic, optical, or other signals capable of being received by communications interface 1524. These signals are provided to communications interface 1524 via a communications path (i.e., channel) 1526. This channel 1526 carries signals and may be implemented using wire or cable, fiber optics, a phone line, a cellular phone link, an RF link, and/or other communications channels.

[00111] In this document, the terms "computer program medium," "computer usable medium," and "computer readable medium" are used to generally refer to media such as main memory 1506 and secondary memory 1512, removable storage drive 1516, a hard disk installed in hard disk drive 1514, and signals. These computer program products are means for providing software to the computer system. The computer readable medium allows the computer system to read data, instructions, messages or message packets, and other computer readable information from the computer readable medium. The computer readable medium, for example, may include non-volatile memory, such as Floppy, ROM, Flash memory, Disk drive memory, CD-ROM, and other permanent storage. It is useful, for example, for transporting information, such as data and computer instructions, between computer systems. Furthermore, the computer readable medium may comprise computer readable information in a transitory state medium such as a

network link and/or a network interface, including a wired network or a wireless network, that allow a computer to read such computer readable information.

[00112] Computer programs (also called computer control logic) are stored in main memory 1506 and/or secondary memory 1512. Computer programs may also be received via communications interface 1524. Such computer programs, when executed, enable the computer system to perform the features of the present invention as discussed herein. In particular, the computer programs, when executed, enable the processor 1504 to perform the features of the computer system. Accordingly, such computer programs represent controllers of the computer system.

CONCLUSION

[00113] Although specific embodiments of the invention have been disclosed, those having ordinary skill in the art will understand that changes can be made to the specific embodiments without departing from the spirit and scope of the invention. The scope of the invention is not to be restricted, therefore, to the specific embodiments. Furthermore, it is intended that the appended claims cover any and all such applications, modifications, and embodiments within the scope of the present invention.

[00114] What is claimed is: